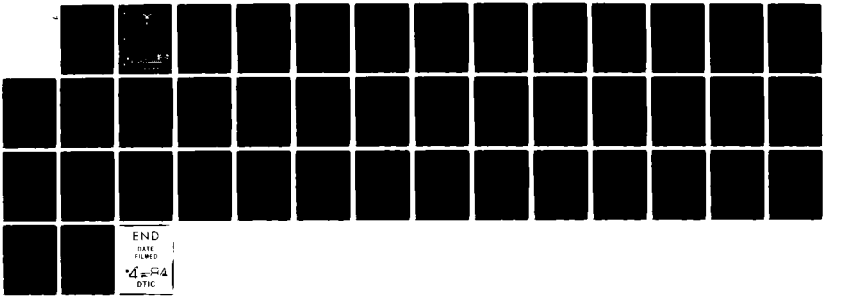
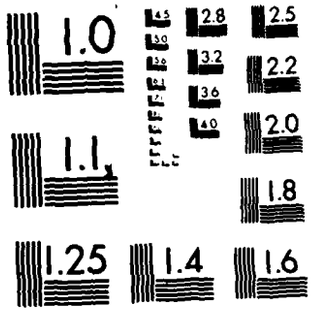
AD-A139 497 VECTOR OPTIMIZATION TECHNIQUES(U) AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOOL OF ENGINEERING A R DEWISPELARE SEP 83 AFIT-EN-TR-83-5 1/1

UNCLASSIFIED F/G 12/1 NL



END
DATE FILMED
4-84
DTIC



VECTOR OPTIMIZATION TECHNIQUES

Technical Report

AU-AFIT-EN-TR-83-5 Aaron R. DeWispelaı

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY (ATC)
AIR FORCE INSTITUTE OF TECHNOLOGY
Wright-Patterson Air Force Base, Ohio

VECTOR OPTIMIZATION TECHNIQUES

Technical Report

AU-AFIT-EN-TR-83-5 Aaron R. DeWispelare

VECTOR OPTIMIZATION TECHNIQUES

AARON R. DEWISPELARE
ASSOCIATE PROFESSOR OF SYSTEMS AND
AEROSPACE ENGINEERING

TECHNICAL REPORT AU-AFIT-EN-TR-83-5
SEPTEMBER 1983

SCHOOL OF ENGINEERING
AIR FORCE INSTITUTE OF TECHNOLOGY
WRIGHT-PATTERSON AIR FORCE BASE, OHIO

ABSTRACT

Multiple Objective Optimization Theory (MOOT) techniques are receiving increasing attention due to their ability to incorporate salient non-commensurate and conflicting objectives of an analysis or design situation into the choice making process. A common implementation of MOOT is by way of a vector optimization process. Vector Optimization is used for generating optimum solutions for alternatives which extremize the components of a vector of objective functions or performance indices. The weighting and constraint techniques are presented as ways of practically implementing an optimization process for a vector of cost functions to generate a Pareto optimal or non-dominated solution set. Computer programs are discussed which accomplish the vector optimization process for the parameter optimization class of linear problems (MOOTLP) and non-linear problems (PROCES). A bibliographic summary of recent vector optimization efforts is included.

TABLE OF CONTENTS

LIST OF SYMBOLS

a_i = i^{th} control/decision variable
a = Vector of control variables

CP = Central processor

LP = Linear program

MOLP = Multiple objective linear program
MOOT = Multiple objective optimization techniques
MOOTLP = Computer program for MOLP formulations

NDSS = Non-dominated solution set
NNDS = Non-dominated solution set (computer produced)

PI = Performance indices
PROCES = Computer program for non-linear vector optimization

SUMT = Sequential unconstrained minimization technique (numerical optimization subroutine)

VOP = Vector optimization problem

W_i = i^{th} weighting coefficient

x_i = i^{th} state variable
x = Vector of state variables

Zi = i^{th} performance index
Z = Vector of performance indices
ZX3LP = IMSL linear programming computer program

I. INTRODUCTION

The ever increasing complexity of decision situations coupled with the requirement to cope with the political, economic, social and technical aspects of these decision situations has resulted in considerable interest being given to implementing new approaches to problem resolution. One such approach which directly considers the importance society places on incorporating the non-commensurate and conflicting objectives of a situation into the choice making process is multiple objective optimization theory (MOOT). Fast and economical computing capability has made the application of MOOT techniques to real world problems, through vector optimization realizable[12].

In MOOT applications, the design or analysis tasks are broken into two parts which allowed for increased efficiency. The modeling and generation of optimal solutions is separated from the preference laden solution selection process. Once a set of generally non-commensurate and competing performance indices (PI) like cost, operational performance, and reliability are established, the engineer or analyst can proceed with the modeling task. This modeling can entail the combination of many subsystem models or submodels which must all be tied back to the PI through a set of model descriptors or state variables. A set of control or decision variables allows for the needed exogenous input. These submodels are generally combined through a computer into an overall system model which is then optimized numerically with respect to the vector of established PI. The solutions generated are "optimal" with respect to the vector of PI. Because no combination of the PI into a scalar needed to be accomplished up to this point, the model and resulting set of

solutions need be accomplished only once, and it is valid for any weighted combination of the PI. For large scale projects, the modeling is an ambitious task, but no more demanding than the traditional engineering approach of scalar optimization. The output of this modeling and vector optimization process is a set of efficient or non-dominated solutions (NDSS). Each solution is defined by its set of state variables, and accompanied by a set of performance index scores. These non-dominated solutions show explicitly the trade-offs among the performance indices for various solution systems as one moves along the efficient solution frontier. Additional sensitivity analysis is generally provided by the optimization software for each efficient solution.

The second major part of the design of analysis process consisting of identifying one solution from the efficient set for development is yet to be accomplished. This identification process results from rank ordering the efficient designs according to a scalar figure of merit which relies on the decision maker's preferences. Each solution in the NDSS is not identifiable as better in its vector form until some sort of preference scheme is applied to select the best solution for the situation. The decision maker decides how important each performance trait is for his situation and then weighs each performance index according to its importance. In addition, the weighting may be increased for those performance values in which there is more confidence or which may be more pertinent to the given situation. The performance indices and weights are combined in a functional form appropriate for the situation [12,17] to allow the computation of a preference score for each solution. The score obtained is a number indicating goodness when compared to other

solutions. The NDSS can be rank ordered by comparing the scores for each member. Thus, the NDSS contains candidate solutions of the given problem. A modification of either the performance indices or the constraints would change the NDSS. However, a modification of the weighting system may change the rank ordering but will not change the members of the NDSS. Non-technical issues, as manifested in the analyst's performance weightings, will determine which NDSS member is the most appropriate for the situation.

Multiple objective optimization techniques have been used successfully in other engineering problems such as missile design [7,9,14], defining a quick response spacecraft [22], aircraft subsystem design [23], Finite Element Modeling [3], and survivability analysis methodology generation [1,2,4].

Section II delineates the vector optimization process along with an example application. Various implementation schemes for a vector optimization process are presented in Section III, followed by discussion of developed software tools which accomplish the vector optimization process for linear systems (MOOTLP) and non-linear systems (PROCES). Section VI presents and example of a MOOT application to missile system design.

II. VECTOR OPTIMIZATION CONCEPTS

Multiple objective optimization theory can be used for generating optimum solutions for the alternative actions which extremize the components of a vector of performance indices. This vector of performance indices or objective functions is optimized with respect to each component of the vector performance index. Many authors [5,8,12,18,24] suggest ways of implementing an optimization process for a vector of objective functions for the following formulation

$$\begin{aligned} &\text{maximize } Z(x) = \\ &\text{maximize } [Z_1(x),\ Z_2(x)\ ,,,,\ Z_p(x)] \end{aligned} \tag{1}$$

$$\text{subject to } g_j(x) \leq 0\ ;\ j = 1,\ 2,\ ,,,\ m \tag{2}$$

$$x_k \geq 0\ ;\ k = 1,\ 2,\ ,,,\ n \tag{3}$$

where $Z(x)$ is a p-dimensional vector of objective functions; i.e., there are p objectives each denoted by Z_i, x is an n dimensional vector of decision variables, and g_j represents the j^{th} constraint on the problem. Since the constraints are of both the equality and inequality type, the number of constraints, m, can be greater than or equal to n without causing the formulation to become overspecified or making the optimization process unnecessary (one unique solution). That is, if s represents the number of equality constraints and q the number of inequality constraints, then $m = q + s$ and we do not violate the appropriateness of the optimization process if $s < n$ and the q inequality constraints are not inconsistent. There is no mathematically based generic definition of "vector optimization" so another

concept of optimality must be employed to identify the "best" or "efficient" set of solutions produced by decision variable values. This concept is generally called Pareto optimality or non-dominance. For the maximization case, a non-dominated solution is signified by the following: for a feasible solution x, there exists no other feasible solution x' such that

$$Z_r(x') > Z_r(x) \quad \text{for some } r = 1, 2, ,,, p \tag{4}$$

$$\text{and } 2_\ell(x') \geq Z_\ell(x) \quad \text{for all } \ell \neq r \tag{5}$$

That is, a specific solution alternative is a non-dominated solution (or Pareto optimal solution) if it is not dominated by another solution alternative. Policy or alternative act A_u dominates policy or alternative A_v if the p vector of performance objectives or attributes for act A_u, which we will call Z^u, is such that each component of the vector, denoted Z_r^u for r = 1,2,,,p, is greater than or equal to (with at least one component strictly greater than) the corresponding component of the performance objective vector for act A_v which we denote Z_r^v.

This concept of Pareto optimality constitutes a process which includes the optimization of vector objective elements combined with a check for dominance to generate a non-dominated solution set (NDSS). There are generally many non-dominated solutions when the objectives are non-commensurate and conflicting in nature. The concept of a NDSS is in some sense analogous to alternate optimal solutions for a scalar objective function. The NDSS forms an "efficient frontier" which represents the best that the system being optimized can do with respect to the vector of performance indices. A

specific NDSS member would then be picked based on some implicit or explicit delineation of a weighting scheme among the individual performance elements. As an illustration of non-domination, consider the two dimensional case with the following solution vectors:

$$Z^1(x) = [Z_1^1(x_1^1) = 1,\ Z_2^1(x_2^1) = 5], \tag{6}$$

$$Z^2(x) = [Z_1^2(x_1^2) = 3,\ Z_2^2(x_2^2) = 6], \tag{7}$$

$$Z^3(x) = [Z_1^3(x_1^3) = 4,\ Z_2^3(x_2^3) = 4]. \tag{8}$$

As can be seen, $Z^2(x)$ dominates $Z^1(x)$, therefore $Z^1(x)$ is eliminated from the NDSS. It can then be determined that $Z^2(x)$ and $Z^3(x)$ are members of the NDSS.

Various authors describe MOOT formulations for popular classes of problems such as deterministic-time invariant, deterministic-time variant, and stochastic-time invariant [5,11].

III. THE CONSTRAINT AND WEIGHTING TECHNIQUES

An implementation problem occurs when one tries to generate this NDSS because of the inability to optimize a vector. Success has been achieved using two techniques to transform the vector of cost functions into a pseudo-scalar optimization form from where normal optimization methods can be applied. These two methods which define, in effect, the implementation of optimization management algorithms for the identification of the NDSS are the constraint technique and the weighting technique.

The Constraint Technique

The constraint technique [5,18,20,23] is formulated to be compatible with the method of proper inequalities. The problem is formulated as follows:

$$\text{Max } Z_f(x) \tag{9}$$

$$\text{subject to } g_j(x) \leq 0 \quad ; j = 1, 2, ,,, m \tag{10}$$

$$x_k \geq 0 \quad ; k = 1, 2, ,,, n \tag{11}$$

$$L_d \leq Z_d \leq U_d \quad ; \text{ for all } d \neq f \tag{12}$$

where U_d is the upper bound and L_d is a lower bound on objective d. Z_f is generally chosen deliberately, and the values of minimum and maximum (L_d, U_d) for each objective d are estimated from rough calculations, expert opinion, or a solution to the scalar optimization problem extremizing only Z_d. Implementation is accomplished by setting all Z_d's equal to an initial set of allowable values of $\overline{Z}_d (Z_d = \overline{Z}_d$, where $\overline{Z}_d$ is a constant; i.e., $L_d \leq \overline{Z}_d \leq U_d)$ and then optimizing to get a value for Z_f. This solution is a possible NDSS. Then the values of $\overline{Z}_d$ are changed and the optimization process iterated to obtain another value for Z_f. This latest set is checked for non-dominance with the first set. This process is continued until all allowable values of $\overline{Z}$

have been used for all objectives d. The NDSS is then presented as the desired output. In summary then, the implementation of the constraint technique converts the vector optimization problem to a sequence of scalar optimization problems shown below

$$\text{Max } Z_f(x) \tag{13}$$

$$\text{subject to } g_j(x) \leq 0 \qquad \forall_j \tag{14}$$

$$x_k \geq 0 \qquad \forall_k \tag{15}$$

$$Z_d = \bar{Z}_d \qquad \forall_{d \neq f} \tag{16}$$

where the formulation is solved repeatedly for various values of $Z_r = \bar{Z}_r$ for each of the r objective elements until the Lagrange multiplier space has been explored for all efficient solutions which are non-dominated.

The Weighting Technique

The weighting technique [5,24] calls for the following formulation

$$\text{Max } J = \text{Max} \sum_{r=1}^{p} w_r Z_r \tag{17}$$

$$\text{subject to } g_j(x) \leq 0 \; ; \; j = 1, 2, ,,, m \tag{18}$$

$$x_k \geq 0 \quad ; \; k = 1, 2, ,,, n \tag{19}$$

where w_r is a scalar weighting coefficient $w_r > 0$ for some $r = 1, 2, ,,, p$ and $w\ell \geq 0$ for all $\ell \neq r$. A convenient convention is to specify $\sum_{r=1}^{p} w_r = 1$. Now J is a scalar function of the objective functions. Implementation is accomplished by varying the parameters w_r over their allowable range (again rough calculations, expert opinion, or scalar optimization for each Z_r can give ranges for w_r's) and optimizing J for each variation. The resulting values of the objectives for each iteration are checked for membership in the NDSS. As with the constraint method, the NDSS is the desired output.

Both the constraint and weighting methods are recognized mathematically for their accuracy in formation of the actual NDSS compared to other techniques which only roughly approximate the NDSS [8] or which require the decision maker to interact periodically during the actual optimization process [5,6,11,15,17]. An obvious advantage of the constraint and weighting techniques is that they can form the actual NDSS without requiring the elicitation of preferences or the interpersonal comparison of these preferences.

Computational considerations involved when one implements either the constraint or weighting technique are discuss in the literature [8,12].

IV. AUTOMATED VECTOR OPTIMIZATION FOR LINEAR FORMULATIONS

(PROGRAM MOOTLP)

This section describes the software tool MOOTLP which is designed to solve multiple objective linear programming problems. The main thrust is to explain the use of this computer programming as it is implemented on the CDC Cyber 175 at Wright-Patterson AFB, Ohio. The program is maintained by the Department of Operational Sciences, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, Ohio. A user's guide is available from the Department of Operational Sciences.

The first task in solving a multiple objective linear programming problem (MOLP) is to identify your problem as a type suitable for this method. There are three main features of a MOLP problem. The first is that there is more than one objective function to be maximized or minimized. (Actually one can use this program to solve problems with only one objective function, but at a loss of some efficiency.) The second is that there are a set of constraints associated with the objective functions. The third feature is that both the objective functions and constraints are linear.

Once the problem has been identified as a MOLP, the next step is to find the solutions. This is a notable difference between MOLP and conventional scalar linear programming problems. For the MOLP problem, there may be a whole set of non-dominated solutions but for the linear programming problems there is generally only one solution. A non-dominated solution set (NNDS) is a set of solutions in which each member neither completely dominates nor is completely dominated by other members of the NNDS.

The final step in a MOLP problem is to analyze the NNDS solution set to rank order the members of the NNDS. Along with finding the non-dominated solution set, this computer program can also be helpful in this final step.

The computer program, MOOTLP, solves MOLP problems by using the constraint technique or the weighting technique. The weighting technique uses a linear combination of the objective functions to form a new objective function and then iteratively solves the new linear programming problem. There are a number of possible solutions because there are several linear combinations used as the new objective function. These solutions are candidates for the NNDS. The constraint technique uses one objective function as the objective function of the new problem and the rest of the objective functions are adjoined as equality constraints with the right hand side equal to one of a range of values appropriate for the objective functions. Again the solutions to these problems are candidates for the NNDS. One characteristic of the weighted technique is that the only solutions found will be corner point solutions (intersections of constraints). The constraint technique on the other hand finds solutions all along the frontier defined by the constraints of the original problem.

The pre-emptive goal programming option in MOOTLP allows desired attainment levels for each objective function to be specified and these attainment levels become additional constraints, in effect. If all these required attainment levels make the problem impossible, then these attainment levels are sequentially relaxed for the objective functions based on the user supplied priorities of the objective functions until a feasible solution is possible. When a feasible solution is possible, the highest priority

objective function is optimized. An alternative goal programming approach using deviational variables can be easily formulated by adjoining these deviational variables with the decision variables in the problem and using either the weighting or constraint technique.

The computer program, MOOTLP, helps analyze the solution set by either weighting the objective functions and then scoring the members of the NNDS to form a ranking among the non-dominated set, or by plotting the NNDS (Option 3). The weighting can be either normalized or regular. For regular weights the score would be computed as follows:

$$\text{Score} = \text{weight}(1) \times Z_1 + \text{weight}(2) \times Z_2 \ldots$$

and for the normalized the score is computed as follows:

$$\text{Score} = W(1) \times Z_1 + W(2) \times Z_2 \ldots$$

where

$$W(i) = \text{weight}(i) / \sum_{i=1}^{n} \text{weight}(i)$$

Types of Problems

This computer program may be used to solve multi-objective, linear programming problems. The current program code will accept a maximum of eight (8) objective functions, ten (10) decision variables, and thirty (30) constraints.

Output

The values of the objective functions, decision variables and dual variables are printed out for each problem. Option 3 (post-process of the NDSS) supplies rankings and graphical output of the NDSS.

Required Problem Format

The actual optimization inside MOOTLP is accomplished by a utility routine (ZX3LP) in the IMSL library [16]. ZX3LP requires all constraints to be less than or equal to ($\leq$) constraints or equality (=) constraints, and ZX3LP also only maximizes. Therefore the MOLP must be formulated as a maximization of objective functions with all constraints either less than or equal to ($\leq$) constraints or equality (=) constraints. Conversion of objective functions which are to be minimized, and greater than or equal to ($\geq$) constraints is accomplished by multiplying through the appropriate equation by a minus one. The objective function(s) and constraint(s) should be formatted such that they contain all the decision variables. Example (given problem contains five decision variables):

$$\text{Min} \quad Z_1 = -5X_1 - 3X_3$$

$$\text{Max} \quad Z_2 = X_2 - X_3 + X_4 - 6X_5$$

S.T.

$$X_1 + X_2 + X_3 - X_4 \geq 10$$

$$X_4 + X_5 \leq 3$$

$$X_1 + X_2 + X_3 = 6$$

should be reformatted as:

$$\text{Max} \quad Z_1 = 5X_1 + 0X_2 + 3X_3 + 0X_4 + 0X_5$$

$$\text{Max} \quad Z_2 = 0X_1 + X_2 - X_3 + X_4 - 6X_5$$

S.T.

$$-X_1 - X_2 - X_3 + X_4 \leq -10$$

$$0X_1 + 0X_2 + 0X_3 + X_4 + X_5 \leq 3$$

$$X_1 + X_2 + X_3 + 0X_4 + 0X_5 = 6$$

The computer algorithm assumes non-negativity constraints involving decision variables. Therefore, do not include these constraints when

formulating problems for input.

Computational Techniques Used

A. Weighting Technique (Option 1)

The weighting technique assigns a weighting factor (w) to each objective function. These weighting factors range according to inputs of the user. The weighting technique finds a set of non-dominated solutions by incrementing all the weighting factors over their input ranges. The range of a weighting factor is divided into intervals according to the number of steps the user requests. For example, for the range of zero to one, if the number of steps equals ten then the weighting factor takes on values from zero to one in increments of .1 units. Required inputs for the weighting technique are as follows:

1. No. of objective functions.
2. No. of decision variables.
3. Objective functions.
4. No. of steps for each weighting factor, range of weights.
5. No. of constraints.
6. Constraints.

The total number of steps for a problem will equal the product of the number of steps for each weighting factor. For example: given three weighting factors ($0 \leq W_1 \leq 1$, $0 \leq W_2 \leq 1$, $0 \leq W_3 \leq 1$) each being incremented by four steps, the total number of steps will equal (4) (4) (4) = 64.

B. Constraint Technique (Option 2)

The constraint technique takes all but one of the objective functions and treats them as constraints. Program MOOTLP treats the first objective function as an objective function while the rest are treated as constraints. By first is meant the first objective function the user inputs. The functions treated as constraints are allowed to range from their lower bound to their upper bound ($lower_i \leq Z_i \leq upper_i$) in the same manner the weighting factors varied over their ranges. Required inputs for the constraint technique are as follows:

1. No. of objective functions.
2. No. of decision variables.
3. Objective functions.
4. No. of steps for each objective function treated as a constraint, range of each objective function treated as a constraint.
5. No. of constraints.
6. Constraints.

The total number of steps will equal the product of the numbers of steps for each objective function acting as a constraint. For example: given three functions ($5 \leq Z_2 \leq 6$, $8 \leq Z_3 \leq 10$, $9 \leq Z_4 \leq 12$) each being incremented by three steps, the total number of steps equals (3) (3) (3) = 27.

Program MOOTLP limits the maximum number of steps for each weighting factor and each objective function treated as a constraint) to 50 steps.

C. Pre-emptive Goal Programming (Option 4)

MOOTLP also solves a pre-emptive goal programming formulation of the MOLP. Minimum acceptable attainment levels (maximization case) for each of the objective functions are specified, as well as the ordered priorities of these objective functions. The program then attempts to solve the MOLP problem by satisfying the specified attainment levels as constraints and then optimizing the objective function with the highest priority. If it is infeasible to satisfy all minimum attainment levels, then the lowest priority objective function's desired attainment level is relaxed and optimization reattempted. This process is repeated until a feasible solution is found. Required inputs for the pre-emptive goal programming option are as follows:

1. No. of objective functions.
2. No. of decision variables.
3. Objective functions.
4. Minimum attainment level desired for each objective function.
5. Priority of each objective function.
6. No. of constraints.
7. Constraints.

WHICH COMPUTATIONAL TECHNIQUE TO USE

The basic difference between the weighting techniques and the constraint technique is the amount of computer processing (CP) time used. The weighting technique generally requires less CP time than the constraint technique. This is because the weighting technique generally finds fewer non-dominated solutions than does the constraint technique. The weighting technique

generates corner-point solutions while the constraint technique may generate solutions anywhere along a frontier of possible non-dominated solutions (including the corner points). The amount of CP time used is affected by the following (in order of importance):

1. Total number of steps.
2. Number of non-dominated solutions found.
3. Number of constraints.

Increasing the total number of steps may enable the computer to find more non-dominated solutions but at the cost of increased CP time. Increasing the number of constraints will increase the computer process time required, but to a lesser degree than if the number of non-dominated solutions found is increased. Since the number of non-dominated solutions found greatly affects computer process time, the exact CP time required for a given problem cannot be determined before actually solving the problem. However, the user may wish to use the following guide to determine which technique to use to satisfy various criteria:

1. If minimum amount of CP time is desired: use weighting technique, Option 1.

2. If minimum amount of computer storage is desired: use weighting technique, Option 1.

3. If corner-point solutions are desired (i.e., just a few solutions): use weighting technique, Option 1.

4. If additional solutions are desired (other than corner-points): use constraint technique, Option 2.

5. If a detailed identification of the NNDS is desired: use constraint technique, Option 2.

6. If specific goals for the objective functions, and priorities for these goals are available; use pre-emptive goal programming, Option 4.

The amount of computer storage used is reflected in the computer process time used, and is affected by the number of non-dominated solutions found. As CP time increases, required storage increases. As the number of solutions found increases, required storage again increases. Program MOOTLP iteratively calls an efficient LP package (ZX3LP) from the IMSL library [16] to perform the optimization.

If a linear formulation is not an appropriate model for the problem, the program PROCES delineated in the next section is designed to handle non-linear formulations.

V. AUTOMATED VECTOR OPTIMIZATION FOR GENERAL NON-LINEAR FORMULATIONS (Program PROCES)

The software tool PROCES solves general non-linear MOOT problems. There is a requirement for increased computer run time as a result of the non-linear numerical techniques used compared to MOOTLP described in Section IV. The operation and use of PROCES are presented for the program as it is currently implemented on the CDC Cyber 175 at Wright-Patterson AFB, Ohio. A user's guide can be obtained from the Department of Operational Sciences, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, who maintain the program.

Program Process

Program PROCES generates the NDSS for a general non-linear, constrained Vector Optimization Problem (VOP). PROCES uses the method of proper equality constraints to find the NDSS. This method converts the VOP to a Scalar Optimization Problem (SOP) by adjoining all but one of the objective functions to the problem as equality constraints. The program iterates through the user selected values for each of the adjoined objective functions. After each iteration, the problem is optimized by SUMT (Sequential Unconstrained Minimization Technique). SUMT is a scalar optimization algorithm that uses the penalty function technique to minimize multivariable, non-linear functions subject to non-linear inequality and equality constraints. The SUMT algorithm is completely documented in the SUMT Version 4 Guide available through DTIC, AD731391 [13,21].

Besides the SUMT algorithm, PROCES consists of the main program and subroutines JCAL, FNSS, and PRINT. The main program controls the program by iterating through all possible combinations of input constrained objective function values. JCAL computes the actual values of the adjoined objective functions based on the x-vector realization calculated by SUMT. FNSS compares each realization of the $\underline{z}$ - vector against the existing NDSS. Dominated solutions are removed from the NDSS. PRINT prints the final NDSS after program execution is completed.

Input Requirements

Data input to PROCES is through the subroutine RESTNT and user created TAPE5. The objective functions and problem constraints are put in RESTNT. RESTNT uses the counter IN to determine which equation will be calculated during each call to the subroutine. TAPE5 contains the VOP parameters and the SUMT options and tolerances. A detailed description of these input requirements is contained in the Execution Procedures section of this guide.

Program Output

All PROCES output is written on one of six tapes. TAPE1 has the SUMT optimization output. This is a detailed description of the optimization. These errors may be the result of an ill-posed problem or certain combinations of values for the constrained objective functions. TAPE2 has the values of every solution that enters the NDSS. It also has the NDSS for each iteration. TAPE6 is the final printout of the NDSS. This tape will contain any information unless the program completes all iterations. TAPE10 contains the current NDSS. Even if the program terminates early, TAPE10 will have the NDSS

on it. If a partial NDSS already exists, it should be written on TAPE10 and the next execution will build on this partial NDSS. TAPE11 is a scratch tape used by FNSS to rewrite the latest NDSS.

An example incorporating PROCES is presented in Section VI.

VI. MOOT EXAMPLE - MISSILE SYSTEM DESIGN

An appropriate application for the MOOT formulation is a technology oriented system such as a missile [7,9,14]. An ideal time period for this effort in the Department of Defense (DOD) Equipment Acquisition Cycle is in the Validation Phase where one or more candidate systems are selected to go into full scale development. This state represents a significant commitment of resources and is analogous to the development preceding a production phase in the private sector. The Validation Phase follows the Conceptual Phase is which a significant "paper study" effort has developed a refined version of the missile in terms of subsystem models. These models are in a condition where they can be formulated into a set of state variable equations and associated constraints which are appropriate for a vector optimization exercise. The Z_i used in this effort are cost of the candidate missile (Z_1), combat effectiveness (Z_2), flight area (Z_3), survivability (Z_4), and reliability (Z_5). The missile is optimized with respect to a vector of these five Z_i. The MOOT process allows the design of efficient candidate missiles (NDSS numbers), in which the tradeoffs associated with each design are evident, without having to prioritize the Z_i at this stage of the effort.

Generic Tactical Missile Description

A generic and relatively inexpensive autonomously operating, conventionally armed, tactical cruise missile has much merit in today's situation of escalating costs of manned military aircraft. Such a missile possessing a guidance unit capable of directing low level terrain following flight, could potentially penetrate air defense systems very successfully.

Since recovery of an unmanned vehicle is not required, its combat range can be much greater than that of a manned aircraft. The successful development of this weapon system depends on the development and integration of several high technology subsystems. These include guidance, sensors, automated target recognition, airframes, navigation algorithms, and armament. It is necessary to model and tune these subsystems for integration into a complete weapons system.

Modeling

The generic missile subsystems were modeled in terms of a set of state equations. The state variables described the "state" of generic missile evaluated. Nine state variables were selected to represent the missile system. These variables are listed below.

x_1 Airframe length (in.)

x_2 Maximum sustainable g's during target run (g).

x_3 Average altitude above ground level (AGL) during target search or attack (ft)

x_4 Fuel flow during target run (lb/sec)

x_5 Fuel capacity (lb)

x_6 Engine length (in)

x_7 Maximum sustainable g's during cruise (g)

x_8 Average altitude AGL during cruise (ft)

x_9 Fuel flow during cruise (lb/sec)

Decision or control variables are those independent variables whose values force the state variables to assume prescribed values, and thus force the

system to assume a certain state. The four decision variables that were used in this effort are listed below:

a_1 Target run Mach number (M)

a_2 Cruise Mach number (M)

a_3 Commanded altitude AGL during target run (ft)

a_4 Commanded altitude AGL during cruise (ft)

Thus the requirement for a generic missile to fly at a given airspeed and altitude will force a particular engine size and fuel flow rate.

State Equations

The state equations relate quantitatively the interaction among the state and decision variables for the generic tactical missile. The state variable realizations in-turn determine the final configuration of the missile subsystems like airframe characteristics (x_1,x_2,x_5,x_6,x_7), and propulsion system characteristics (x_4,x_5,x_6,x_7,x_8,x_9). The state equations for the generic missile are given below:

$$x_1 = 104.5 + 0.11*x_5 + x_6$$

$$x_2 = 6.79*a_1*((9.85*10^{-3}*x_6 + 0.11)/a_1^2 - 0.58)^5$$

$$x_3 = a_3 + 143.35*a_1 - 21.08*x_2 + 23.52$$

$$x_4 = 7.53*10^{-3}/a_1^2 + 0.21*a_1^2$$

$$x_5 = (7.71 \times 10^3*x_9)/a_2 + (12.35*x_4)/a_1$$

$$x_6 = 2.21/a_2^2 + 64.48*a_2^2 + 2.0$$

$$x_7 = 5.85*a_2^2*((1.05*10^{-2}*x_6 + 0.12)/a_2^2 - 0.04)^5$$

$$x_8 = a_4 + 88.41*a_2 - 12.05*x_2 + 27.43$$

$$x_9 = 0.01/a_2^2 + 0.24*a_2^2$$

The coefficients and functional forms of these non-linear equations were derived from standard missile equations of motion models and verified using a six-degree-of-freedom simulation [14]. This missile simulation was validated using actual wind tunnel and flight test data of simular missiles.

Resource Constraints

The next set of expressions to be developed were those which represent physical or operational constraints upon the generic missile system, its subsystems, or system employment in terms of an expected scenario. These constraints serve the dual purpose of keeping the optimization focused on those potential solutions which are physically feasible, and reducing the processing time to find a solution by limiting the problem space. The resource constraints are as listed below:

$$119 \leq x_1 \leq 150$$

$$0.1 \leq x_2 \leq 3.0$$

$$a_3 \leq x_3 \leq 4.0$$

$$0 \leq x_4$$

$$500.0 \leq x_5 \leq 1320.0$$

$$31.0 \leq x_6 \leq 48.0$$

$$1.0 \leq x_7 \leq 2.1$$

$$a_4 \leq x_8 \leq 4.0$$

$$0 \leq x_9$$

$$0.4 \leq a_1 \leq 0.65$$

$$0.2 \leq a_2 \leq 0.6$$

$$10. \leq a_3 \leq 30.$$

$$20. \leq a_4 \leq 70.$$

These constraints can be interpreted respectively as airframe size limitations based on proposed launcher systems (x_1), flight restrictions in low altitude maneuvers ($x_2,x_3,x_7,x_8,a_1,a_2,a_3,a_4$), positive fuel flow ($x_4,x_9$), fuel capacity based on airframe restrictions (x_5), and engine size based on thrust requirements and airframe restrictions(x_6).

Performance Indices

The Z_i used were functions of the state and decision variables. Utilizing such tools as cost estimating relationships, empirical performance relationships, and regression techniques, the coefficients and functional form for each of the Z_i were determined.

Cost (Z_1) was a function of the various subsystems included in the missile. Cost estimating relations were used in conjunction with the state variables to produce estimates for research, development, testing, evaluating, production, operation, and support costs in terms of a constant and functional relationship to state variable, x_6. The units of Z_1 are monetary.

Combat Effectiveness (Z2) provides a measure of the expected value of enemy equipment damaged. Combat effectiveness was primarily a function of flight altitude, velocity, and sustainable g's. The units of Z_2 are normalize value units.

Flight Area (Z_3) provides a combined measure of range covered in flight. This measure was non-dimensionalized by dividing the mission minimum required range. Z_3 was a function of velocity and fuel usage.

Reliability (Z_4) is an indication of the probability that the missile will function properly under established environmental conditions for the duration of the mission. Based on historical and projected data, constant failure

rates were used for the components of the generic missile. A series arrangement of critical subsystems was used to model the component functioning. No redundancy of components was allowed due to volume limitations. The resulting reliability estimates were relatively insensitive to state and decision variables because all missile flight times were short compared to component failure times.

Survivability (Z_5) measures in a relative fashion, the survivability of the generic missiles when performing the mission in the presence of enemy defense systems. The survivability factor is a function of velocity, flight altitude, and maximum sustainable g's. Mathematic equations for the Z_i are as follows:

$$Z_1 = (1.06*10^3 + ((x_6 - 31.0)/12.1*139.9)*10^3$$

$$Z_2 = (3.05*x_2*a_1^2)/(34.75*a_3^4 + 1.0)$$

$$Z_3 = 0.18*(x_5 - 147.06*x_9/a_2*a_1/x_4)$$

$$Z_4 = 0.96*\exp(- x_2/a_1*x_7/a_2*10^{-3})$$

$$Z_5 = (1.1*a_1 - 0.44)*(- 1.0*10^{-3}*x_3 + 0.53)*(0.06*x_2 + 0.18)$$

Computer Implementation

Now that the generic missile has been modeled in terms of a MOOT formulation

$$\text{Ext}Z(x) = [\min Z_1, \max Z_2, \max Z_3, \max Z_4, \max Z_5]$$

such that

$$g_j(x) \leq 0 \quad \forall_j \quad \text{(general constraints)}$$

$$h_i(x) = 0 \quad \forall_i \quad \text{(state equations)}$$

the task of optimizing each candidate design and then comparing the results arises. The above formulation is implemented on the digital computer utilizing the "constraint technique" described in Section III. An executive program manages the vector optimization process, categorizes solutions, and saves the non-dominated solutions in terms of the state and control variable values. The actual optimization is performed by a subroutine which uses the Sequential Unconstrained Minimization Technique (SUMT) as its theoretical basis [13]. The non-linearities present in the Z_i's and constraints require a numerical optimization technique like SUMT version 4 [21] for an efficient solution.

SUMT is the optimization routine (Fig. 1) that is iteratively called by the main program-PROCES. After the constraint technique is used to reformulate the vector optimization problem into the pseudo scalar optimization problem of eqns 13-16, this formulation is transformed into an unconstrained penalty function. This new problem now takes the unconstrained form of:

$$P = Z_f - r \cdot \sum_{j=1}^{m} \ln(g_j) + \frac{1}{r} \cdot \sum_{i=1}^{q} h_i^2 + \frac{1}{r} \cdot \sum_{d=1}^{p-1} (Z_d - Z_d)^2 \qquad (20)$$

where Z_f is one of the original performance measures, (g_j) are the inequality constraints, (h_i) are the equality constraints, Z_d is the value that the dth performance measure must attain, r is a monotonically decreasing positive constant, and P is the penalty function. As the minimization problem numerically approaches optimum, r is decreased, increasing the effects of the penalties until, under suitable conditions, P approaches Z_f. When this occurs, the problem solution has converged.

The requirements placed on the model by SUMT were that there exist a feasible convex region so that optimization can take place. Because the state

equations ($h_i(x) = 0$) were not linear, a global extremism could not be guaranteed, but analysis revealed that a global solution was indeed found. The Lagrange Multipliers which are supplied by SUMT were used to check the Kuhn-Tucher necessary conditions as well as being useful in a sensitivity analysis exercise.

The program, PROCES (Fig.1) was implemented on a CDC-CYBER 175 system in FORTRAN 4. The program including SUMT Version 4 was about 3000 lines, required 60K of CM to execute, and converged to a single NDSS in an average time of approx. 20 CPU sec. The long run times were caused by the non-linear forms of Z_i and constraints which slowed the convergence process considerably. All the subroutines in SUMT are called by program PROCES except GRAD, OPT, REVALU, PUNCH, and XMOVE. The user supplied subroutines used were READIN and RESTNT. Newton's method was used by SUMT to perform the optimization.

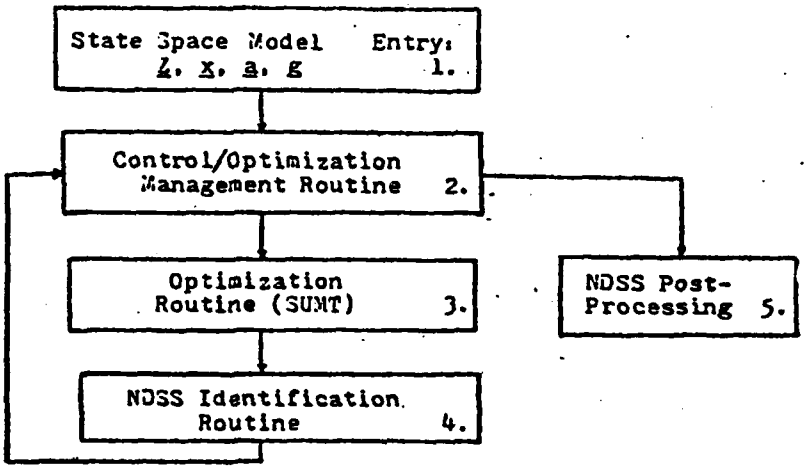


Figure 1. Program PROCES, Vector Component Optimization Algorithm

Figure 1. Program Flow of PROCES

Results

The MOOT formulation when implemented on the computer produced the NDSS comprised of the seven efficient designs shown in Table 1.

The trade-offs for the best of all candidates is efficiently presented in this tabular NDSS. It is interesting to note that the Reliability and Survivability are fairly constant for all candidates indicating that the type and length of mission effected all candidates the same. There is some variability in Flight Area and much variability in Cost and Combat Effectiveness.

If it is not possible for the DM to make his choice with the NDSS and proceed to the production process, it is apparent that the incomplete ordering of this NDSS could be converted to a complete ordering by forming a scalar scoring function or figure of merit [5,17]. The DM's value system could then bring about a ranking of the members of the NDSS.

One popular way to form a scalar scoring function is to produce a weighted sum of the Z_i. This scalar takes the form of:

$$S = \sum_i W_i Z_i \tag{21}$$

where W_i is the relative weight of the ith performance index of Z(x), S is the score of a member of the NDSS corresponding to Z(x). For instance, if the DM's preferences for cost (Z_1) and combat effectiveness (Z_2) were twice as important as the other indicies, then a set of weights could be $W_1 = W_2 = .286$, and $W_3 = W_4 = W_5 = 0.143$. Scoring each of the NDSS members in Table 1 produces a ranking of scores ranging from 2.998 to 1.733 with candidate member 7 topping the list.

Program MOOTLP has a NDSS post-process routine called PREANA which interactively forms this ranking for the DM after the NDSS is identified.

Because the NDSS will not change unless the model changes (Z_i's, constraints etc.) it needs to be identified through the vector optimization process only once. If the preferences of the DM change in the selection process, it is a simple task to enter new weights and rerank the members of the NDSS to identify the candidate system with which to continue development.

Table 1

	Z_1	Z_2	Z_3	Z_4	Z_5
NDSS	Cost	Combat Effectiveness	Flight Area	Reliability	Survivability
1.	1.937	2.393	2.400	.946	.618
2.	1.690	2.501	2.314	.744	.631
3.	1.667	2.494	2.312	.940	.617
4.	1.700	2.363	2.429	.945	.621
5.	2.855	5.241	2.906	.956	.616
6.	2.854	5.531	2.609	.955	.618
7.	2.856	5.602	2.479	.954	.627

VII. SUMMARY

Vector optimization implementations of MOOT are receiving increased use in the design and analysis arena as their efficiencies are realized when compared to traditional engineering analysis and design which utilizes scalar optimization. In this regard, three significant observations can be made.

1. The computer implementation of the MOOT formulation allows a complex and non-linear problem model to be solved numerically. The computer program PROCES finds mathematically efficient designs in terms of a set of constraints and state equations to optimize the vector Z. The resulting solutions are compared quantitatively in terms of elements of Z, to determine which belong to the "efficient set" or NDSS. This output NDSS presents trade-offs for many efficient designs (not just one) with respect to all elements of Z simultaneously. This NDSS can be readily converted to a numerical ranking if this is deemed necessary by the DM.

2. This formulation has inherent flexibility in that changes can be made in the formulation and the effects of these changes can be seen for all candidate solutions rapidly by executing the vector optimization process. For instance, if a binding constraint is changed significantly, the new NDSS could be generated again fairly easily by simply executing the program PROCES. This allows current information to be included into the model at all stages of the project thereby minimizing the modeling and analysis ramifications. This flexibility extends to the subjective aspects of the modeling effort (DM's preferences which modify constraints and utility for various attributes) as well as objective aspects (hardware oriented state equations and state-of-the-

art development and constraint limitations).

3. Once the set of efficient candidate solutions has been formed (NDSS), this set will not change unless the model changes. The ranking of the members of the NDSS and subsequent selection of a system from among the NDSS can be accomplished with very little effort in a post-processing routine. Since this ranking depends on the preference of the DM, any change in the preference structure requires only a reranking of the same NDSS and not a rerunning of the optimization program to form a new NDSS. The separation of the decision process into an objective part (the modeling and optimization to identify the NDSS) and a subjective part (the establishing of the preferences of the DM to rank the members of the NDSS) is especially appealing to both analyst and DM.

The paper demonstrates that MOOT can be a very valuable tool for a computer aided design of hardware oriented systems. The technique is an efficient way to formulate and solve the vector (multiple performance measures) optimization problem and as such is an excellent alternative to traditional scalar optimization. Through the resultant non-dominated solution set it also provides an excellent tool with which to do a trade-off analysis, both among candidates and for the solutions of a single candidate. MOOT allows the analyst to observe and evaluate a system's design in terms of the broad spectrum of needs that the system must satisfy. MOOT can be used to model very complex systems with a high degree of accuracy. However, processing time and storage requirements increase with the more complex models.

BIBLIOGRAPHY

1. Bednarz, E.J., DeWispelare, A.R. "State Space Model Concept for Evaluating Survivability Methodologies for Aircraft Design," Proceedings of the AIAA 9th Annual Mini-Symposium on Aerospace Science and Technology; Air Force Institute of Technology, Wright-Patterson AFB, OH, 1983.

2. Bednarz, E.J., et.al. "A Procedure to Evaluate Survivability Methodologies for Aircraft Design," Vols I-IV, AD-B071109/10/11/099, GSE Design Study; School of Engineering, Air Force Institute of Technology; Wright-Patterson AFB, OH, 1982.

3. Briggs, H.C., DeWispelare, A.R. "Application of Multiple Objective Optimization Techniques to Finite Element Model Tuning," Proceedings of the 24th Structural Dynamics and Materials Conference; Lake Tahoe, NV, 1983.

4. Bubb, K.W., DeWispelare, A.R. "Vector Optimization Applied to Survivability Methodology Evaluation," Proceedings of the AIAA 9th Annual Mini-Symposium on Aerospace Science and Technology; Air Force Institute of Technology, Wright-Patterson AFB, OH, 1983.

5. Chankong, V., Haimes, Y. Multiobjective Decision Making, North Holland Publishing Co., New York 1983.

6. Charnes, A., Cooper, W. W. Management Models and Industrial Applications of Linear Programming, Vol. I, John Wiley, New York, 1961.

7. Clark, D. et.al. "Conceptual Design of an Advanced Strategic Air Launched Missile", Vols. I-III, GSE Design Study; School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, 1983.

8. Cohon, J.L., Marks, D.M. "A Review and Evaluation of Multiobjective Programming Techniques," Water Resources Research, Vol. II, No. 2, 1975.

9. DeWispelare, A.R. "A Computer Based Application of Non-Linear Multiple Objective Optimization," Journal on Computers and Industrial Engineering, Vol. 2, 1984.

10. DeWispelare, A.R. "Multiple Objective Optimization Applied to Missile System Design," Proceedings of the Fourth International Conference on Mathematical Modeling; Zurich, Switzerland, 1983.

11. DeWispelare, A.R., Sage, A.P. "On Combined Multiple Objective Optimization and Multiple Attribute Utility Theory for Evaluation and Choice Making," Journal of Large Scale Systems, No. 2, 1981.

12. DeWispelare, A.R. "Algorithm Efficiency in Generating Non-Dominated Solution Sets," Proceeding of the IEEE 12th Annual Symposium on Systems Theory, 1980.

13. Fiacco, A.V., McCormick, G.P. Nonlinear Programming: Sequential Unconstrained Minimization Techniques, John Wiley and Sons, New York, N.Y., 1968.

14. Gibson, R.S., et. al. "Investiagation of the Feasibility of a Conventionally Armed Tactical Cruise Missile," Vol. II, GSE Design Study; School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, 1981(2).

15. Haimes, Y.Y., Hall, W.A., Freedman, A.M. Multiobjective Optimization in Water Resource Systems, Elsiver, 1975.

16. IMSL, International Mathematical and Statistical Libraries, 7500 Bellaire Boulevard, Houston, Texas, 1979.

17. Keeney, R.L., Raiffa, H. Decisions With Multiple Objectives: Preferences and Value Tradeoffs, Wiley, New York, 1976.

18. Lin, J.C. "Proper Inequality Constraints and Maximization of Index Vectors," Journal of Optimization Theory and Applications, Vol. 21, No. 4, 1977.

19. MacCrimmon, K.R. "An Overview of Multiple Objective Decision Making," in Multiple Criteria Decision Making, J.L. Cochrane and M. Zeleny (ed.), University of South Carolina Press, Columbia, South Carolina, 1973.

20. Marglin, S.A., Public Investment Criteria, MIT Press, Cambridge, Massachusetts, 1967.

21. Mylander, W.C. "A Guide to SUMT-Version 4", AD-731391; Research Analysis Corp., McLean, VA, 1971.

22. Robinson, D.G., et.al. "Conceptual Feasibility Study of An Advanced Military Spaceflight Capability," Vols I-III, AD-C027170/1/2, GSE Design Study; School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, 1981(1).

23. Tabak, D. "Numerical Aspects of Multicriteria Optimization in Systems Design," Proceedings IEEE, Systems, Man, and Cybernetics International Conference, Tokyo, Japan, 1978.

24. Zadeh, L.A. "Optimality and Non-Scalar-Valued Performance Criteria," IEEE Transactions Automatic Control, AC-8(1), 1963.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

AD A139 477

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION: nclassified	1b. RESTRICTIVE MARKINGS:
2a. SECURITY CLASSIFICATION AUTHORITY:	3. DISTRIBUTION/AVAILABILITY OF REPORT: Approved for public release - distribution unlimited.
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE:	
4. PERFORMING ORGANIZATION REPORT NUMBER(S): AU-AFIT-EN-TR-83-5	5. MONITORING ORGANIZATION REPORT NUMBER(S):
6a. NAME OF PERFORMING ORGANIZATION: School of Engineering; 6b. OFFICE SYMBOL (If applicable): AFIT/ENY	7a. NAME OF MONITORING ORGANIZATION:
6c. ADDRESS (City, State and ZIP Code): Air Force Institute of Technology, Wright-Patterson AFB, OH 45433	7b. ADDRESS (City, State and ZIP Code):
8a. NAME OF FUNDING/SPONSORING ORGANIZATION: ; 8b. OFFICE SYMBOL (If applicable):	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER:
8c. ADDRESS (City, State and ZIP Code):	10. SOURCE OF FUNDING NOS. — PROGRAM ELEMENT NO.: ; PROJECT NO.: ; TASK NO.: ; WORK UNIT NO.:
11. TITLE (Include Security Classification): Vector Optimization Techniques	
12. PERSONAL AUTHOR(S): Aaron R. DeWispelare	
13a. TYPE OF REPORT: Technical Report; 13b. TIME COVERED: FROM _____ TO _____	14. DATE OF REPORT (Yr., Mo., Day): September 1983; 15. PAGE COUNT: 35
16. SUPPLEMENTARY NOTATION:	
17. COSATI CODES — FIELD: ; GROUP: ; SUB. GR.:	18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number): Optimization, Vector Optimization, Computer Application Numerical Analysis, Missile Design, Pareto Optimal Mathematical Program

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

Multiple objective Optimization Theory (MOOT) techniques are receiving increasing attention due to their ability to incorporate salient non-commensurate and conflicting objectives of an analysis or design situation into the choice making process. A common implementation of MOOT is by way of a vector optimization process. Vector Optimization is used for generating optimum solutions for alternatives which extremize the components of a vector of objective functions or performance indices. The weighting and constraint techniques are presented as ways of practically implementing an optimization process for a vector of cost functions to generate a Pareto optimal or nondominated solution set. Computer programs are discussed which accomplish the vector optimization process for the parameter optimization class of linear problems (MOOTLP) and non-linear problems (PROCES). A bibliographic summary of recent vector optimization efforts is included.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT: CLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ DTIC USERS ☐	21. ABSTRACT SECURITY CLASSIFICATION: Unclassified
22a. NAME OF RESPONSIBLE INDIVIDUAL: PETER J. TORVIK	22b. TELEPHONE NUMBER (Include Area Code): (513) 255-3069; 22c. OFFICE SYMBOL: AFIT/ENY

DD FORM 1473, 83 APR EDITION OF 1 JAN 73 IS OBSOLETE.

SECURITY CLASSIFICATION OF THIS PAGE